REPLACEMENT CLAIMS FOR SPECIFICATION

The big implementation issue here is the following: plugins cannot be changed dynamically in nessusd except through a restart. And which service is running can be known only after running find_services, hence it has to be done in two steps, and store and recover the KB in between the two runs.

Run 1)

Load port_scanner_equivalent and find_services, UPD or TCP as is the case. Launch the plugins and populate the KB with the appropriate attribute depending on the service found.

Save the KB.

Run 2)

Read the KB, and depending on the service, find out which plugins have to be loaded. For instance if the attribute filled up is WWW/.services then all .nasl plugins which have WWW/services in the script_require_ports will have to be loaded.

Then, have all the plugins in the plugin directory and restart nessusd with the appropriate plugin directory at the same time recovering the KB.

Then run those plugins.

I claim

1. A system for real-time vulnerability assessment of a host/device, said system comprising:

an agent running on the host/device, said agent comprising:

a first data structure for storing the status of interfaces and ports on the interfaces of the host/device,

an executable agent module coupled to the first data structure to track the status of interfaces and ports on the interfaces of the host/device and to store the information, as entries in said first data structure,

said executable agent module to compare the entries to determine a change in the status of interfaces and/or of ports on the interfaces of the host/device,

a remote destination server, said destination server comprising,

a second data structure for storing the status of interfaces and the ports on the interfaces of the host/device,

The big implementation issue here is the following: plugins cannot be changed dynamically in nessusd except through a restart. And which service is running can be known only after running find_services, hence it has to be done in two steps, and store and recover the KB in between the two runs.

Run 1)

Load port_scanner_equivalent and find_services, UPD or TCP as is the case. Launch the plugins and populate the KB with the appropriate attribute depending on the service found.

Save the KB.

Run 2)

Read the KB, and depending on the service, find out which plugins have to be loaded. For instance if the attribute filled up is WWW/.services then all .nasl plugins which have WWW/services in the script_require_ports will have to be loaded.

Then, have all the plugins in the plugin directory and restart nessusd with the appropriate plugin directory at the same time recovering the KB.

Then run those plugins.

I claim

1. A system for real-time vulnerability assessment of a host/device, said system comprising:

an agent running on the host/device, said agent comprising:

a first data structure for storing the status of interfaces and ports on the interfaces of the host/device,

an executable agent module coupled to the first data structure to track the status of interfaces and ports on the interfaces of the host/device and to store the information, as entries in said first data structure,

said executable agent module to compare the entries to determine a change in the status of interfaces and/or of ports on the interfaces of the host/device,

a remote destination server, said destination server comprising,

a second data structure for storing the status of interfaces and the ports on the interfaces of the host/device,

an executable server module coupled to the second data structure to receive the information communicated by the agent executable module of the agent on the host/device,

said executable server module to store the received information as entries in the second data structure wherein the entries indicate the state of each of the ports on each of the active interfaces of the host/device as received,

said executable server module to compare the entries in said data structures to determine the change in the status of interfaces and ports on the interfaces of the host/device, and

said executable server module to run vulnerability assessment tests on the host/device in the event of a change in the status of interface/ports.

2. The system of claim 1, further comprising:

an executable server module coupled to a second data structure to receive and update the vulnerability data in the destination server used by the server for vulnerability tests, whenever new vulnerabilities are discovered, and

said executable server module coupled to the second data structure to test the host/device for the new vulnerabilities whenever the vulnerability database is updated with new vulnerabilities and to determine the new vulnerabilities

3. A system for real-time vulnerability assessment of a host/device, said system comprising:

an agent running on the host/device, said agent comprising:

a first data structure to store the status of interfaces on the host/device and the ports on the interfaces on the host/device,

an executable agent module coupled to the first data structure and operable to track the status of interfaces and ports on the interfaces of the host/device to collect and store the information, as entries in the first data structure,

said executable agent module coupled to the first data structure to compare the entries to determine a change in the status of interfaces and/or of ports on the interfaces of the host/device,

said executable agent module to communicate said changes to a remotely located destination server on the network, and

a destination server running remotely, said destination server comprising:

a second data structure for storing the status of interfaces/ports on the host/device,

an executable server module coupled to the second data structure to receive information communicated by the executable module on the host/device,

said executable server module coupled to the second data structure to store the received information as entries in the second data structure wherein the entries indicate the state of each of the ports on each of the active interfaces of the host/device as received,

said executable server module coupled to the second data structure to compare the entries to determine any change in the status of interfaces and ports on the interfaces of the host/device as reported to it,

said executable server module coupled to the second data structure to process the changes to determine any new interfaces active and/or any newly opened ports on any of the active interfaces on the host/device on which services are listening as reported to it,

said executable server module coupled to the second data structure to run tests remotely to identify the network services running on the newly opened ports on the various active interfaces of the host/device,

said executable server module coupled to the second data structure to run vulnerability assessment tests on the identified network services on the newly opened ports of the interfaces and storing the results, and

said executable server module coupled to the second data structure to obtain an incremental or an overall vulnerability status report of the host/device from the results of the current vulnerability tests, and previously stored results.

4. The system of claim 3, further comprising:

an executable server module coupled to the second data structure to receive and update the vulnerability database in the vulnerability assessment server used by the server to do vulnerability tests, whenever new vulnerabilities are discovered publicly or elsewhere, and

an executable server module coupled to the second data structure to test the host/device for the new vulnerabilities whenever the vulnerability database is updated with new vulnerabilities, and obtain results.

5. The system of claims 1 and 4, wherein status of an interface is either active or inactive.

6. The system of claims 1 and 4, wherein status of a port is a service listening on the port or not.
7. The system of claims 1 and 4, wherein the agent tracks the change in status of ports/interface by monitoring in real-time or polling at periodic intervals for the status of ports/interfaces and storing the entries at various time intervals.
8. The system of claims 1 and 4, wherein the communication protocol between the host/device and the destination server is a standard transport level utility selected from sockets or any other standard communication protocol.
9. The system of claims 1 and 4, wherein the server executable module compares the entries corresponding two consecutive time intervals.
10. The system of claims 1 and 4, wherein the host/device is selected from a switch, a router, a device running a standard real-time operating system, a mobile device or a PDA.
11. The system of claims 1 and 4, wherein the host/device is an enterprise/consumer machine running with Windows, Unix, Linux, VxWorks, Symbian or PalmOS.
12. The system of claims 1 and 4, wherein the changes that are communicated to the destination server consisting of the IP address of the interface(s) and the port numbers on which listening services have started or stopped on the particular interface(s).
13. The system of claims 1 and 4, wherein the status of the port consists of separate statuses for TC and UD protocols.
14. The system of claims 1 and 4, wherein plurality of hosts/devices is tracked in conjunction with one or more destination servers handling the host/devices.
15. Logic encoded in media for real-time vulnerability assessment of a host/device, and operable to perform the following steps:
    a) tracking in real-time the status of interfaces and/or of the ports on a host/device,
    b) communicating a change in the status of the interfaces and/or the status of ports of the host/device to a remotely located destination server on the network,
    c) tracking in real-time the reported status of ports and interfaces of the host/device by the destination server, and
    d) conducting vulnerability assessment tests on the host/device by the destination server in the event of a change in the status of interfaces and/or ports of the host/device.

16. Logic encoded in media for real-time vulnerability assessment of a host/device, and operable to perform the following steps:
    a) tracking in real-time the status of interfaces and/or ports on a host/device,
    b) communicating the change in the status of the interfaces and/or the status of ports to a remotely located destination server on the network,
    c) tracking in real-time the reported status of the ports and interfaces of the host/device by the destination server,
    d) processing the changes by the destination server to determine new active interfaces or newly opened ports on any of the active interfaces on the host/device on which services are listening,
    e) running tests to identify remotely the network services running on the newly opened ports on the various active interfaces of the host/device,
    f) running vulnerability assessment tests on the identified network services on the newly opened ports of the interfaces and storing the results, and
    g) generating an incremental and/or overall vulnerability status report of the host/device from the results of the current vulnerability tests, and storing the results classified port and interface wise
17. The logic of claims 15 and 16, wherein the status of an interface is either active or inactive.
18. The logic of claims 15 and 16, wherein status of a port is a service listening on the port or not.
19. The logic of claims 15 and 16, wherein the status of the port consists of separate statuses for TC and UD protocols.
20. The logic of claims 15 and 16, wherein tracking consists of monitoring in real-time or polling at periodic intervals for the status of ports/interfaces on the host/device.
21. The logic of claims 15 and 16, wherein the communication protocol between the host/device and the destination server is a standard transport level utility selected from sockets or any other standard communication protocol.
22. The logic of claims 15 and 16, wherein the host/device is selected from a switch, a router, a device running a standard real-time operating system, a mobile device or a PDA.
23. The logic of claims 15 and 16, wherein the host/device is an enterprise/consumer machine running with Windows, Unix, Linux, VxWorks Symbian or PalmOS.

24. The logic of claims 15 and 16, wherein the changes that are communicated to the destination server consisting of the IP address of the interface(s) and the port numbers on which listening services have started or stopped on the particular interface(s).

25. The logic of claims 15 and 16, wherein the information that is communicated from the host/device to the destination server is the names of the services.

26. The logic of claims 15 and 16, wherein the information that is communicated from the host/device to the destination server is a message signaling a change in the status of interfaces and/or ports on the host/device.

27. The logic of claims 15 and 16, wherein the vulnerability assessment server used by the destination server is updated with the new vulnerabilities to test the presence of vulnerabilities.

28. The logic of claims 15 and 16, wherein a plurality of hosts/devices are tracked in conjunction with plurality of destination servers handling the host/devices.

29. A computer-implemented method for real-time vulnerability assessment of a host/device, said method comprising:

a) tracking in real-time the status of interfaces and ports on the host/device,

b) collecting and storing the status as entries in a data structure,

c) comparing the entries to determine any change in the status of interfaces and/or the status of ports on the interfaces of the host/device,

d) communicating the changes to a remotely located destination server on the network,

e) storing said changes as entries in a data structure by the destination server wherein the entries indicate the state of each of the ports on each of the active interfaces of the host/device as reported,

f) comparing the entries by the destination server to determine if there is any change in the status of interfaces and ports on the interfaces of the host/device as reported to it, and

g) running vulnerability assessment tests on the host/device by the destination server and reporting the results.

30. A computer-implemented method for real-time vulnerability assessment of a host/device, said method comprising:

a) polling the status of the ports and interfaces on the host/device, periodically at a pre-configured time interval,

b) collecting the above information and storing as entries in the first data structure of an agent,

c) comparing the entries to determine if there is any change in the status of interfaces and/or the status of ports on the interfaces of the host/device,

d) communicating the changes to a remotely located destination server on the network,

e) storing the received information as entries in the second data structure of a server by the destination server wherein the entries indicate the state of each of the ports on each of the active interfaces of the host/device as reported,

f) comparing the entries by the destination server to determine if there is any change in the status of interfaces and ports on the interfaces of the host/device as reported to it, and

g) running vulnerability assessment tests on the host/device by the destination server and reporting the results.

31. The method of claims 29 and 30, wherein the status of an interface is either active or inactive.

32. The method of claim 29 and 30, wherein the status of a port is a service listening on the port or not.

33. The method of claim 29 and 30, wherein the agent tracks the change in status of ports/interface by monitoring in real-time or polling at periodic intervals for the status of ports/interfaces and storing the entries at various time intervals.

34. The method of claim 29 and 30, wherein the communication protocol between the host/device and the destination server is a standard transport level utility selected from sockets or any other standard communication protocol.

35. The method of claim 29 and 30, wherein the server executable module compares the entries corresponding two consecutive time intervals.

36. The method of claim 29 and 30, wherein the changes that are communicated to the destination server consisting of the IP address of the interface(s) and the port numbers on which listening services have started or stopped on the particular interface(s).

37. The method of claim 29 and 30, wherein the status of the port consists of separate statuses for TC and UD protocols.

38. The method of claim 29 and -30, wherein plurality of hosts/devices is tracked in conjunction with one or more destination servers handling the host/devices.